Ontology-based Text Document Clustering

Andreas Hotho and Alexander Maedche and Steffen Staab
Institute AIFB, University of Karlsruhe, 76128 Karlsruhe, Germany
{aho, ama, sst}@aifb.uni-karlsruhe.de
http://www.aifb.uni-karlsruhe.de/WBS

Abstract

Text clustering typically involves clustering in a high dimensional space, which appears difficult with regard to virtually all practical settings. In addition, given a particular clustering result it is typically very hard to come up with a good explanation of why the text clusters have been constructed the way they are. In this paper, we propose a new approach for applying background knowledge during preprocessing in order to improve clustering results and allow for selection between results. We preprocess our input data applying an ontology-based heuristics for feature selection and feature aggregation. Thus, we construct a number of alternative text representations. Based on these representations, we compute multiple clustering results using K-Means. The results may be distinguished and explained by the corresponding selection of concepts in the ontology. Our results compare favourably with a sophisticated baseline preprocessing strategy.

1 Introduction

With the abundance of text documents available through the Web or corporate document management systems, the dynamic partitioning of document sets into previously unseen categories ranks high on the priority list for many applications like business intelligence systems. However, current text clustering approaches tend to neglect several major aspects that greatly limit their practical applicability.

First, text document clustering is mostly seen as an *objective* method, which delivers one clearly defined result, which needs to be "optimal" in some way. This, however, runs contrary to the fact that different people have quite different needs with regard to clustering of texts because they may view the same documents from completely different perspectives (e.g., a business view vs. a technical view; also cf. (Macskassy et al., 1998)). Thus, what is needed are document clustering methods that provide multiple *subjective* perspectives onto the same document set.

Second, text document clustering typically is a machine learning task taking place in a *high-dimensional space* of word vectors, where each word, i.e. each entry of a vector, is seen as a potential attribute for a text. Empirical and mathematical analysis, however, has shown that – in addition to computational ineffiencies – clustering in high-dimensional spaces is very difficult because every data point tends to have the same distance from all other data points (cf. (Beyer et al., 1999)).

Third, text document clustering *per se* is often rather useless, unless it is combined with an *explanation* of why particular texts were categorized into a particular cluster. I.e. one output desired from clustering in practical settings is the explanation of why a particular cluster result was produced besides of the result itself. A common method for producing explanations is the learning of rules based on the cluster results. Again, however, this approach suffers from the high number of features chosen for computing clusters.

Though there are of course different approaches for clustering, simple ones like K-Means or sophisticated ones (like (Bradley et al., 1998)), based on the consideration just mentioned we found that virtually all algorithms working on large feature vectors will eventually face the same principal problems regarding *high-dimensional space* without really approaching the matters of *subjectivity* and *explainability*. Therefore, our objective has been the consideration of different views of the data, i.e. different representations[1] of the same set of text documents, from which alternative clustering results may be derived.

The principal idea of our approach, COSA (Concept Selection and Aggregation), is based on the usage of a simple, core ontology for generating alternative representations of the given document set such that from the various representations multiple clustering result may be constructed by standard

[1] Motivated by the database point of view, we also call derived text representations "aggregations".

clustering algorithms like K-Means. The single representations are construed by aggregating the original term vector representation in various ways. More precisely, we have compiled a heterarchy of concepts[2]. The heterarchy is navigated top-down by COSA in order to select document features (i.e. concepts) for an aggregated vector representation. Thereby, COSA considers that features are neither too frequent (i.e. COSA would split them into their subconcepts) nor too infrequent (i.e. COSA would abandon them in favor of more frequent ones) to be meaningful for clustering.

Thus, a set of clustering results is produced without interaction by a human user of the system. The user may then decide to prefer the one over the other clustering result based on the actual concepts used for clustering as well as on standard quality measures (such as the silhouette measure (Kaufman & Rousseeuw, 1990)).

In this paper, we first formalize our notion of ontology (Section 2). Then, we describe COSA as well as two baseline preprocessing strategies (Section 3). These three preprocessing strategies constitute the basis for our application of K-Means in the experimental evaluation of Section 4. From our experiments we have derived some informal experiences that we describe in our "Lessons Learned" Section.

2 Heterarchy and Core Ontology

A core ontology in our framework is defined by:

Definition 1 (Core Ontology) *A core ontology is a sign system* $\boldsymbol{O} := (\mathcal{L}, \mathcal{F}, C^*, \mathcal{H}, \text{ROOT})$, *which consists of*

- ***A lexicon:*** *The lexicon* $\mathcal{L}$ *contains a set of natural language terms.*
- *A set of* ***concepts*** C^*.
- *The* ***reference function*** $\mathcal{F}$ *with* $\mathcal{F}: 2^{\mathcal{L}} \mapsto 2^{C^*}$. $\mathcal{F}$ *links sets of terms* $\{L_i\} \subset \mathcal{L}$ *to the set of concepts they refer to. In general, one term may refer to several concepts and one concept may be refered to by several terms. The inverse of* $\mathcal{F}$ *is* $\mathcal{F}^{-1}$.
- ***A heterarchy*** $\mathcal{H}$***:*** *Concepts are taxonomically related by the directed, acyclic, transitive, reflexive relation* $\mathcal{H}$, ($\mathcal{H} \subset C^* \times C^*$).
- *A top concept* ROOT $\in C^*$. *For all* $C \in C^*$ *it holds*: $\mathcal{H}(C, \text{ROOT})$.

Example 1 (Example Ontology)

- ***lexicon*** $\mathcal{L}$ = {Hotel, Grand Hotel, Hotel Schwarzer Adler, Accommodation, ...)
- ***concepts*** C^* = {ROOT, HOTEL, ACCOMMODATION, …}
- ***reference function*** $\mathcal{F}$ = {(Hotel, HOTEL), (Grand Hotel, HOTEL), (Hotel Schwarzer Adler, HOTEL), …}, i.e. *"Hotel", "Grand Hotel" and "Hotel Schwarzer Adler" refer to the concept* HOTEL .
- ***heterarchy*** $\mathcal{H}$ = {(HOTEL, ACCOMMODATION), (ACCOMMODATION, ROOT), …}

The core ontology defines the background knowledge used for preprocessing and selection of relevant views (i.e. aggregations) onto the set of texts. The formulation we have used here roughly corresponds to the basic structures used in the famous WordNet (Miller, 1995), but the actual ontology we have used is domain-specific rather than general as WordNet.

[2] A heterarchy of concepts is a kind of "taxonomy" where each term may have multiple children and multiple parents

3 Document Preprocessing

Documents may be represented by a wide range of different feature descriptions. The most straight-forward description of documents relies on term vectors. A term vector for one document specifies how often each term from the document set occurs in that document. The immediate drawback of this approach for clustering is the size of the feature vectors. In our example evaluation, the feature vectors computed by this method were of size 46,947, which made clustering inefficient and difficult in principle, as described above.

While for supervised learning tasks there exist quite a number of evaluations of how document preprocessing strategies perform (cf., e.g., (Fuernkranz et al., 1998)), there are only few corresponding results for unsupervised knowledge discovery tasks like document clustering (cf. Section 6).

To evaluate our approach, which takes advantage of the background knowledge we provide with our core ontology, we have compared against that approach for document preprocessing (referred to by *Simple Vector Representation* or *SiVeR* in the following). We were aware that due to the problems with clustering in high dimensional space, SiVeR would be handicapped from the very beginning. In order to perform a more competitive comparison, we have decided to include another preprocessing approach in the evaluation.

Hence, in the following we develop, *(i)*, a preprocessing strategy (cf. Section 3.1) based on term vectors reduced to terms considered "important" by information retrieval measures, *viz*. a preprocessing strategy based on term selection; *(ii)*, a more comprehensive approach using the background knowledge available in the ontology. In particular, we apply techniques from natural language processing to map terms to concepts (cf. Section 3.2) and we select between various aggregations navigating top-down in the heterarchy.

3.1 Preprocessing Strategy: Term Selection (TES)

Term selection, the second approach we use here for preprocessing, is based on the feature vectors from SiVeR, but focuses on few terms, hence, it produces a low dimensional representation. Selection of terms is based on the information retrieval measure *tfidf*:

Definition 2 (*tfidf*) *Let tf(i,j) be the term frequency of term j in a document* $d_i \in D^*$, *i= 1,...,N. Let df(j) be the document frequency of term j that counts in how many documents term j appears. Then tfidf* (*term frequency / inverted document frequency*) *of term j in document is defined by*:

$$tfidf(i,j) = tf(i,j) * \log\left(\frac{N}{df(j)}\right) \tag{1}$$

Tfidf weighs the frequency of a term in a document with a factor that discounts its importance when it appears in almost all documents. Therefore terms that appear too rarely or too frequently are ranked lower than terms that hold the balance and, hence, are expected to be better able to contribute to clustering results.

For TES, we produce the list of all terms contained in one of the documents from the corpus D^* except of terms that appear in a standard list of stopwords. Then, TES selects the *dim* best terms *j* that maximize *W(j)*,

$$W(j) = \sum_{i=1}^{N} tfidf(i,j) \tag{2}$$

and produces a *dim* dimensional vector for document d_i containing the *tfidf* values, *tfidf*(*i*,*j*) for the *dim* best terms.

3.2 Preprocessing Strategy: Concept Selection and Aggregation (COSA)

Our approach for preprocessing, concept selection and aggregation (COSA), involves two stages. First, COSA maps terms onto concepts using a shallow and efficient natural language processing system. Second, COSA uses the concept heterarchy to propose good aggregations for subsequent clustering.

3.2.1 Mapping Terms to Concepts

The mapping of terms to concepts in our approach relies on some modules of SMES (Saarbrücken Message Extraction System), a shallow text processor for German (cf. (Neumann et al., 1997)). SMES components exploited by COSA comprise a *tokenizer* based on regular expressions and a *lexical analysis* component including a *word* and a so-called *domain lexicon* (the domain specific part of the lexicon partially defines $\mathcal{F}$).

The tokenizer scans the text in order to identify boundaries of words and complex expressions like "$20.00" or "United States of America", and to expand abbreviations. The word lexicon contains more than 120,000 stem entries. Lexical analysis uses the word lexicon, *(i)*, to perform morphological analysis of terms, i. e. the identification of the canonical common stem of a set of related word forms and the analysis of compounds and, *(ii)*, to recognize named entities. Thus, $\mathcal{L}$ as described in Definition 1 is a set defined by the tokenizer, the word lexicon and the analysis procedures of the lexical analysis component. The domain lexicon contains the mappings from word stems to concepts, i.e. together with the other modules it represents the function $\mathcal{F}$ as defined in Definition 1. By this way, e.g., the expression "Hotel Schwarzer Adler" is associated with the concept HOTEL.During the mapping process we do not resolve ambiguities of terms. This means, if we find several concepts with the same lexical entry we map the term to all related concepts. Based on this input, each document is represented by a vector of concepts, each entry specifying
the frequency that a concept occurs in the document.

3.2.2 A heuristic for generating good aggregations

Because synonyms are mapped to common concepts and because in all realistic document sets there are more terms than concepts, the size of concept vectors representing documents is already considerably smaller than the size of term vectors produced by SiVeR. Still, realistic settings require at least some hundreds or thousands of concepts, which yields simply too many dimensions for practical clustering and explantion.

Therefore, we have looked for heuristics to further reduce the number of features. The principal idea of our algorithm GENERATECONCEPTVIEWS lies in navigating the heterarchy top-down splitting the concepts with most *support* (cf. (4)) into their subconcepts and abandoning the concepts with least support. (cf. Algorithm 1 below). Thus, the algorithm generates lists of concepts that appear neither too often nor too rarely. The rationale is that too (in-)frequent concept occurences are not appropriate for clustering.

$$Support(i,C) := \sum_{\substack{B \in C^* \\ H(B,C)}} cf(i,B) \tag{3}$$

$$Support(C) := \sum_{i=1}^{N} Support(i,C) \tag{4}$$

The variable *Agenda* is defined to describe the current list of concepts used to generate a particular representation from the given document set. For instance, the current *Agenda* could be [ACCOMODATION, VACATION, SIGHT-SEEING]. An aggregation is altered, by taking the frontmost, i.e. the concept with the most support, from the agenda (lines 4 and 5) and branching – if it is not a

leaf concept – into its subconcepts (line 10). In order to restrict branching, we only perform binary splits at a time. Continuing the example just given, the first feature for the input space is described by the concept ACCOMODATION and when ACCOMODATION has the subconcepts [HOTEL, GUEST-HOUSE, YOUTH-HOSTEL], we will select the immediate subconcept that has the highest support of these three (line 11), e.g. HOTEL, and aggregate the other two subconcepts into one feature, viz. [GUEST-HOUSE, YOUTH-HOSTEL] (line 12). The list [GUEST-HOUSE, YOUTH-HOSTEL] is then treated almost like a proper atomic concept. HOTEL and [GUEST-HOUSE, YOUTH-HOSTEL] are both inserted into *Agenda* ordering all elements according to their support (lines 13-14). The result might be, e.g., [VACATION, [GUEST-HOUSE, YOUTH-HOSTEL], HOTEL, SIGHT-SEEING].

Thereby, *direct support* of a concept C in a document d_i is defined by the concept frequency $cf(i,C)$ that one of the terms $\mathcal{F}^{-1}(\{C\})$ appears in d_i (cf. (3)). Complete support includes also consideration of all the subconcepts (cf. (3)).

If the *Agenda* has length *dim* + 1 due to the last binary split of one of its elements, *Agenda* is shortened by the element with least support (line 15). If the *Agenda* has the correct number of features, it is added to the output set describing a selection of concepts, hence an aggregation that represents documents by *dim*-dimensional concept vectors (line 17).

Thus, Algorithm 1 zooms into those concepts that exhibit strongest support, while taking into account the support of subconcepts. Finally, it proposes sets of aggregations for clustering that imply a *dim*-dimensional representation of documents by concept vectors. Each entry of a vector specifies how often the concept (or its subconcepts) appears in the corresponding document.

3.3 A note on absolute vs. logarithmic values and normalized vectors

The document representations described so far use absolute frequency values for concepts or terms (possibly weighted by *idf*). Considering that the occurrence of terms forms a hyperbolic distribution and, hence, most terms appear only rarely, using the logarithmic value `log(x+1)` instead of the absolute value `x` itself seemed reasonable to improve clustering results. Indeed, for all preprocessing strategies given here, we found that results were only improved compared to absolute values. Hence, all results presented subsequently assume the logarithmic representation of term or concept frequencies. Furthermore, we compared normalized vector representations against absolute or logarithmic values. For this latter comparison, we could not find any interesting differences, with respect to our measure (cf. sec. 4).

Algorithm 1 (GENERATECONCEPTVIEWS)
Input: number of dimensions dim, Ontology $\mathcal{O}$ *with top concept* ROOT *document set* D^*

```
1  begin
2     Agenda := [ROOT];
3     repeat
4           Elem := First(Agenda);
5           Agenda := Rest(Agenda);
6           if Leaf(Elem)
7                 then continue := FALSE;
8                 else
9                       if Atom(Elem) then Elem := Subconcepts(Elem); fi;
10                      NewElem := BestSupportElem(Elem);
11                      RestElem := Elem \ NewElem;
12                      if Not Empty(RestElem) then Agenda := SortInto(RestElem, Agenda); fi;
13                      Agenda := SortInto(NewElem, Agenda);
14                      if Length(Agenda) > dim then Agenda := Butlast(Agenda); fi;
15          fi;
16          if Length(Agenda) = dim then Output(Agenda); fi;
17    until continue = FALSE;
18 end
```

Output: *Set of lists consisting of single concepts and lists of concepts, which describe feature selections corresponding to different representations of the document corpus D^*.*

Auxiliary functions used:

Subconcepts(*C*)	*returns an arbitrarily ordered list of direct subconcepts of C.*
Support(*C*)	*cf. equation 4.*
Support(*ListC*)	*is the sum over all concepts C in ListC of Support*(*C*)*.*
SortInto(*Element, List2*)	*sorts Element , which may be a single concept or a list of concepts, as a whole into List2 ordering according to Support(Element) and removing redundant elements.*
BestSupportElem(*List*)	*returns the Element of List with maximal Support(Element).*
[*Element*]	*constructs list with one Element.*
[*Element, List*]	*list constructor extending List such that Element is first.*
First(*List*), *Rest*(*List*)	*are the common list processing functions.*
Atom(*E*)	*returns true when E is not a list.*
Leaf(*E*)	*returns true when E is a concept without subconcepts.*
List \ E	*removes element E from List.*
Length(*List*)	*returns the length of List.*
Butlast(*List*)	*returns a list identical to List, but excluding the last element.*

4 Evaluation

This section describes the evaluation of applying K-Means to the preprocessing strategies SiVeR, TES, and COSA introduced above.

Setting

We have performed all evaluations on a document set from the tourism domain (cf. (Klettke et al., 2001)). For this purpose, we have manually modeled an ontology $\mathcal{O}$ consisting of a set of concepts C ($|C| = 1030$), and a word lexicon consisting of 1950 stem entries (the coverage of different terms $\mathcal{L}$ by SMES is much larger!). The heterarchy $\mathcal{H}$ has an average depth of 4.6, the longest uni-directed path from root to leaf is of length 9.

Our document corpus D^* has been crawled from a WWW provider for tourist information (URL: http://www.all-in-all.de) consisting now of 2234 HTML documents with a total sum of over 16 million terms. The documents in this corpus describe actual objects, like locations, accomodations, facilities of accomodations, administrative information, and cultural events.

Because of our aim of producing *multiple subjective* clustering results, it was difficult to compare *objectively*, e.g. with a uniquely categorized set of documents. Therefore we have resorted to a mathematical evaluation of the clustering results only - which, of course, is only possible for approaches that work in the same or a comparable representation space. Thus, our aim was to compare SiVeR, TES, and COSA – and not, e.g., Latent Semantic Indexing – for a wide range of parameter settings.

Silhouette Coefficient

In order to be rather independent from the number of features used for clustering and the number of clusters produced as result, our main comparisons refer to the silhouette coefficient (cf. (Kaufman & Rousseeuw, 1990)):

Definition 3 (Silhouette Coefficient) *Let* $D_M = \{\overline{D}_1,...,\overline{D}_k\}$ *describe a clustering result, i.e. it is an exhaustive partitioning of the set of documents* D^**. The distance*[3] *of a document* $d \in D^*$ *to a cluster* $\overline{D}_i \in D_M$ *is given as*

$$dist(d,\overline{D}_i) = \frac{\sum_{p \in \overline{D}_i} dist(d,p)}{\left|\overline{D}_i\right|} \tag{5}$$

Let further be $a(d,D_M) = dist(d,\overline{D}_i)$ the distance of document *d* to its cluster $\overline{D}_i (d \in \overline{D}_i)$, and $b(d,D_M) = \min_{\overline{D}_i \in D_M, d \notin \overline{D}_i} dist(d,\overline{D}_i)$ the distance of document *d* to the nearest neighbouring cluster.

The silhouette $\vartheta(d,D_M)$ of a document *d* is then defined as:

$$s(d,D_M) = \frac{b(d,D_M) - a(d,D_M)}{\max\{a(d,D_M), b(d,D_M)\}}. \tag{6}$$

The silhouette coefficient as:

$$\mathrm{SC}(D_M) = \frac{\sum_{p \in D^*} s(p,D_M)}{\left|D^*\right|}. \tag{7}$$

The silhouette coefficient is a measure for the clustering quality, that is rather independent from the number of clusters, *k*. Experiences, such as documented in (Kaufman & Rousseeuw, 1990), show that values between 0.7 and 1.0 indicate clustering results with excellent separation between clusters, *viz*. data points are very close to the center of their cluster and remote from the next nearest cluster. For the range from 0.5 to 0.7 one finds that data points are clearly assigned to cluster centers. Values from 0.25 to 0.5 indicate that cluster centers can be found, though there is considerable "noise", i.e. there are many data points that cannot be clearly assigned to clusters. Below a value of 0.25 it becomes practically impossible to find significant cluster centers and to definitely assign the majority of data points.

For comparison of the three different preprocessing methods we have used standard K-Means[4]. However, we are well aware that for high-dimensional data approaches like (Bradley et al., 1998) may improve results – very likely for all three preprocessing strategies. However, in preliminary tests we found that in the low-dimensional realms where the silhouette coefficient indicated reasonable separation between clusters, quality measures for standard and improved K-Means coincided.

The general result of our evaluation using the silhouette measure was that K-Means based on COSA preprocessing excelled the comparison baseline, *viz*. K-Means based on TES, to a large extent. K-Means based on SiVeR was so strongly handicapped by having to cope with overly many dimensions that its silhouette coefficient always approached 0 – indicating that no reasonable clustering structures could be found.

One exemplary, but overall characteristic diagramm depicted in Figure 1 shows the silhouette coefficient for a fixed number of features used (namely 15) and a fixed number of clusters produced (namely 10). It does so for K-Means based on SiVeR, for K-Means based on TES, and for K-Means based on COSA. The results for SiVeR are strictly disappointing. TES is considerably better, but it still yields a silhouette coefficient that indicates practically non-existent distinctions between clusters. COSA produces for this parameter setting 89 aggregations. We found that the best aggregations produced from COSA delivered clustering results with silhouette measures of up to 0.48 – indicating indeed very reasonable separation between clusters. The second part of figure 1 shows the corresponding MSE (means square error) value for every aggregation and for TES.

[3] We use the standard euclidean distance for computing the silhouette coefficient.
[4] We use several well know heuristics to derive a good starting solution for K-Means.

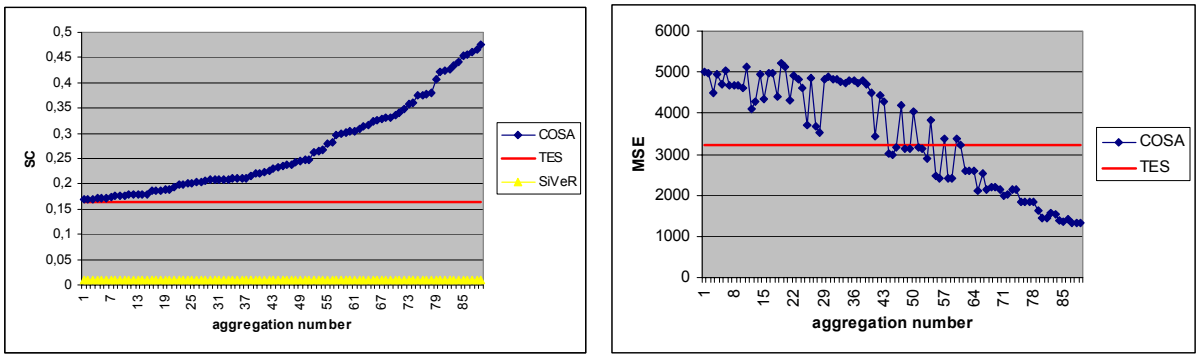

Figure 1: Comparing TES with 89 aggregations produced by COSA for k =10; *dim* = 15.

Varying number of features *dim* and clusters *k*

Then we explored how COSA and TES would fare when varying the number of features used and the number of clusters produced by K-Means.

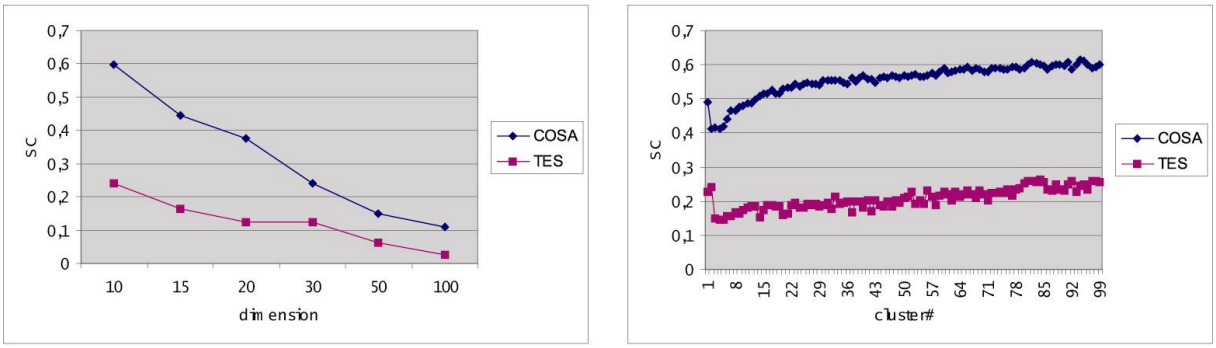

Figure 2: a) Comparing TES and the best aggregation of COSA; k =10; *dim* = 15= 10, 15, 30, 50, 100.
b) Comparing TES and the best aggregation produced by COSA; k = 2…100; *dim* = 15.

Figure 2a depicts the dependency between the number of features, *dim*, used and the preprocessing method for a fixed number of clusters, viz. k = 10. The line for COSA shows the silhouette coefficient for the best aggregation from the ones generated by GENERATECONCEPTVIEWS. We see that for TES and COSA the quality of results decreases – as expected – for the higher dimensions (cf. (Beyer et al., 1999)), though COSA still compares favorably against TES.

We have not included the lower bound of COSA in Figure 2a. The reason is that – so far – we have not been very attentive to optimize GENERATECONCEPTVIEWS in order to eliminate the worst aggregations up front. This, however, should be easily possible, because we observed that the bad results are produced by aggregations that contain too many overly general concepts like MATERIALTHING or INTANGIBLE.

In our real-world application we experienced that it is useful to include the users viewpoint for deriving the number of dimensions with respect to the actual problem. In general one may propose the following upper bound for the number of useable dimensions: The silhouette coefficient decreases below 0.25 using more than 30 dimensions. Thus, using more than 30 dimensions may not be useful, because no meaningful clustering structure may be discovered.

In the next experiments, we have varied the number of clusters, k, between 2 and 100, while *dim* remained at 15 (cf. Figure 2b). The general result is that the number of clusters does not affect the results produced by COSA and TES very much. The silhouette coefficient grows slightly with the number of clusters, because of a growing number of documents that cluster exactly at one point.

Example for Interpretation

In order to provide a more concrete intuition of the type of results returned by GENERATECONCEPTVIEWS, we here show the list of concepts that corresponds to the best aggregation for parameters $k = 10$ and $dim = 10$ and a silhouette coefficient of 0.598:

```
SAUNA, SOLARIUM, TERRACE, BEACH, SEA_RESSORT, ACTION_AT_OBJECT,
    OVERNIGHT_STAY, WATER_SPORTS, TRAVELING, HOTEL_CATEGORY
```

Comparing some plain lists may already give the user an intuition of how clustering results might be distinguishable (or not distinguishable if the aggregations are very similar!). A better grip at interpretation is however achieved by depicting the relevant parts of the heterarchy as shown in Figure 3.

Here, one may recognize that NONPRIVATE_FACILITIES_OF_ACCOMODATION and ACTIONS were important concepts used for clustering and distinguishing documents. Interpreting results, we may conjecture that HOTEL_CATEGORY (three, four, five star hotels, etc.) is a concept, which might perhaps correlate with facilities of the accomodation – a correlation that happens to be not described in the given ontology. Finally, we see SEA RESSORT in this aggregation, which might play a role for clustering or which might occur just because of uninterpretable "noise".

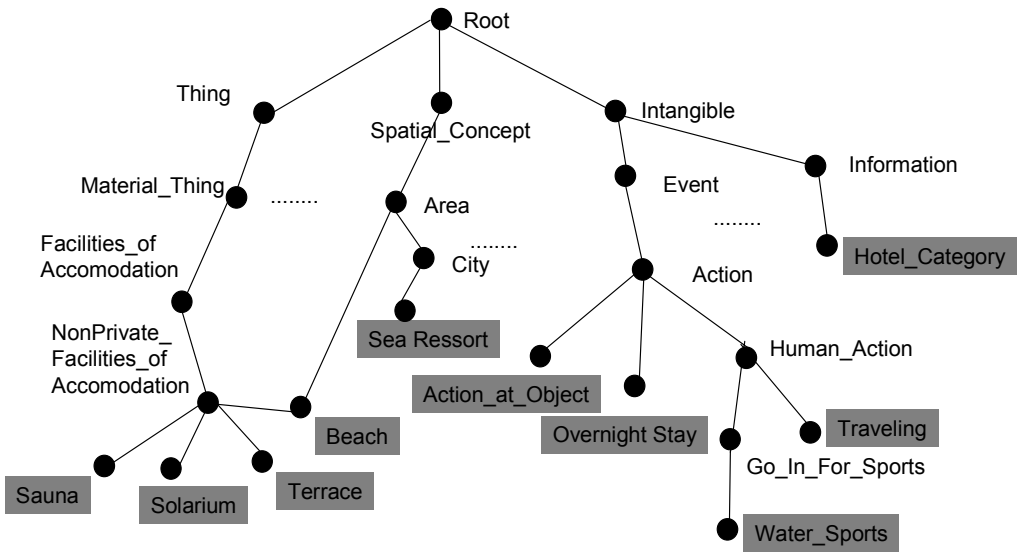

Figure 3: An example aggregation generated by COSA

We currently explore GUI possibilities in order to tie the interpretation of clustering results with the navigation of the heterarchy in order to give the user a good grip at different clustering aggregations.

5 Lessons Learned

From our experiments we have learned quite a number of interesting lessons that may be summarized as follows. First of all, a number of expectations we had, e.g., based on our literature research were fulfilled:

- Clustering in high-dimensional space is worse than in fewer dimensions.
- Clustering results produced in high-dimensional space are hard to interpret for humans. Every effort at producing interpretations from there seemed to require projections, aggregation or any other techniques for reducing the dimensions in the end – even though these techniques may incur a loss of information.

- The labelling of clustering results with selected concepts seems to facilitate the interpretation task for the user. Though we have not performed usability studies to prove this claim, we believe our experiences are strong enough to warrant plausibility.

Some results were not at all obvious from the beginning:

- Aggregations into a low-dimension space do not improve clustering *per se*. Bad evaluation results have been due to aggregations with too many overly general concepts (like MATERIAL_THING or INTANGIBLE).

- Aggregations based on leaf of near-leaf concepts yield good evaluation results. For these aggregations we find that a substantial share of documents are represented by $\vec{0}$, a concept vector where all entries are 0. The reason is that in particular aggregations, i.e. seen from a particular part of the ontology, this share of documents is simply irrelevant.

- Analyzing single clusters that stem from COSA clustering results with the silhouette coefficient, we find clusters that are very well separated from the rest. However, quite regularly we find clusters that cannot be separated from the rest with good quality. We assume that texts in such clusters can only be reasonably interpreted in alternative views.

Our results support the general statement that structure can mostly be found in a low dimensional space (cf. (Beyer et al., 1999)). Our proposal is well suited to provide a selected number of aggregations in subspaces exploiting standard K-Means and comparing favorably with baselines, like clustering based on *dim* terms ranked by *tfidf* measures. The selected concepts may be used to indicate to the user, which text features were most relevant for the particular clustering results and to distinguish different aggregations.

6 Related Work

All clustering approaches based on frequencies of terms/concepts and similarities of data points suffer from the same mathematical properties of the underlying spaces (cf. (Beyer et al., 1999; Hinneburg et al., 2000)). These properties imply that even when "good" clusters with relatively small mean squared errors can be built, these clusters do not exhibit significant structural information as their data points are not really more similar to each other than to many other data points. Therefore, we derive the high level requirement for text clustering approaches that they either rely on much more background knowledge (and thus can come up with new measures for similarity) or that they cluster in subspaces of the input space.

In general, existing approaches (e.g., (Agrawal et al., 1998; Hinneburg & Keim, 1999)) on sub-space clustering face the dual nature of "good quality". On the one hand, there are sound statistical measures for judging quality. State-of-the-art methods use them in order to produce "good" projections and, hence, "good" clustering results, for instance:

- Hinneburg & Keim (Hinneburg & Keim, 1999) show how projections improve the effectiveness and efficiency of the clustering process. Their work shows that projections are important for improving the performance of clustering algorithms. In contrast to our work, they do not focus on cluster quality with respect to the internal structures contained in the clustering.

- The problem of clustering high-dimensional data sets has been researched by Agrawal et al. (Agrawal et al., 1998): They present a clustering algorithm called CLIQUE that identifies dense clusters in subspaces of maximum dimensionality. Cluster descriptions are generated in the form of minimized DNF expressions.

- A straightforward preprocessing strategy may be derived from multivariate statistical data analysis known under the name principal component analysis (PCA). PCA reduces the number of features by replacing a set of features by a new feature representing their combination.

- In (Schuetze & Silverstein, 1997), Schuetze and Silverstein have researched and evaluated projection techniques for efficient document clustering. They show how different projection techniques significantly improve performance for clustering, not accompanied by a loss of cluster quality. They distinguish between local and global projection, where local projection maps each document onto a different subspace, and, global projection selects the relevant terms for all documents using latent semantic indexing (introduced by (Deerwester et al., 1990)).

Now, on the other hand, in real-world applications the statistically optimal projection, such as used in the approaches just cited, often does not coincide with the projection most suitable for humans to solve a particular task, such as finding the right piece of knowledge in a large set of documents. Users typically prefer explicit background knowledge that indicates the foundations on which a clustering result has been achieved.

Hinneburg et al. (Hinneburg et al., 1999) consider this general problem a domain specific optimization task. Therefore, they propose to use a visual and interactive environment to derive meaningful projections involving the user. Our approach may be seen to automatically solve some part of the task they assign to the user environment, while giving the user some first means to explore the result space interactively in order to select the projection most relevant for her particular objectives.

Finally, we want to mention an interesting proposal for feature selection made in (Devaney & Ram, 1998). Devaney and Ram describe feature selection for an unsupervised learning task, namely conceptual clustering. They discuss a sequential feature selection strategy based on an existing COBWEB conceptual clustering system. In their evaluation they show that feature selection significantly improves the results of COBWEB. The drawback that Devaney and Ram face, however, is that COBWEB is not scalable like K-Means. Hence, for practical purposes of clustering in large document repositories, COSA seems better suited.

7 Conclusion

In this paper we have shown how to include background knowledge in form of a heterarchy in order to generate different clustering aggregations from a set of documents. We have compared our approach against a sophisticated baseline, achieving a result favourable for our approach. In addition, we have shown that it is possible to automatically produce results for diverging views of the same input. Thereby, the user can rely on a heterarchy to control and possibly interpret clustering results.

The preprocessing method, COSA, that we propose is a very general one. We have applied our techniques on a high-dimensional data set that is not based on text documents, but on a real-world customer database with 24,156 customers in the telecommunications domain Preliminary results of applying our method on this complex transaction-oriented database show similar positive results as could be presented here for text clustering.

Further work will also compare our results with an other low dimensional baseline taken from latent semantic indexing.

References

Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (1998). Automatic subspace clustering of high dimensional data for data mining applications. In *Proceedings of SIGMOD-1998*, pages 94-105. ACM Press.

Beyer, K., Goldstein, J., Ramakrishnan, R., & Shaft, U. (1999). When is `nearest neighbor' meaningful. In *Proceedings of ICDT-1999*, pages 217-235.

Bradley, P., Fayyad, U., & Reina, C. (1998). Scaling clustering algorithms to large databases. In *Proceedings. of KDD-1998*, pages 9-15. AAAI Press.

Deerwester, S. C., Dumais, S. T., Landauer, T. K., Furnas, G. W., & Harshman, R. A. (1990). Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391- 407.

Devaney, M. & Ram, A. (1998). Efficient feature selection in conceptual clustering. In *Proceedings of ICML-1998t*, pages 92-97. Morgan Kaufmann.

Fuernkranz, J., Mitchell, T., & Riloff, E. (1998). A Case Study in Using Linguistic Phrases for Text Categorization on the WWW. In *Proc. of AAAI/ICML Workshop Learning for Text Categorization*, pages 5-12. AAAI Press.

Hinneburg, A., Aggarwal, C., & Keim, D. (2000). What is the nearest neighbor in high dimensional spaces? In *Proceedings of VLDB-2000*, pages 506-515. Morgan Kaufmann.

Hinneburg, A. & Keim, D. A. (1999). Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. In *Proceedings of VLDB-99*, pages 506-517. Morgan Kaufmann.

Hinneburg, A., Wawryniuk, M., & Keim, D. A. (1999). Hd-eye: visual mining of high-dimensional data. *IEEE Computer Graphics and Applications*, 19(5):22-31.

Kaufman, L. & Rousseeuw, P. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley, New York.

Klettke, M., Bietz, M., Bruder, I., Heuer, A., Priebe, D., Neumann, G., Becker, M., Bedersdorfer, J., Uszkoreit, H., Maedche, A., Staab, S., & Studer, R. (2001). GETESS - Ontologien, objektrelationale Datenbanken und Textanalyse als Bausteine einer Semantischen Suchmaschine. *Datenbank-Spektrum*, 1(1), 14-24.

Macskassy, S. A., Banerjee, A., Davison, B., & Hirsh, H. (1998). Human performance on clustering web pages: a preliminary study. In *Proceedings of KDD-1998*, pages 264-268. AAAI Press.

Miller, G. (1995). WordNet: A lexical database for english. *CACM*, 38(11):39-41.

Neumann, G., Backofen, R., Baur, J., Becker, M., & Braun, C. (1997). An information extraction core system for real world german text processing. In *Proceedings of ANLP-1997*, pages 208-215.

Schuetze, H. & Silverstein, C. (1997). Projections for efficient document clustering. In *Proceedings of SIGIR-1997*, pages 74-81. Morgan Kaufmann.

Andreas Hotho is a Ph.D. student at the Institute of Applied Computer Science and Formal Description Methods at Karlsruhe University. He earned his Master's Degree in information systems from the University of Braunschweig (Germany) in 1998. His research interests include the application of data mining and machine learning techniques on very large databases, text mining, semantic web mining and intelligent web applications .

Dr. Alexander Maedche is head of the research department WIM (Knowledge Management) at the FZI Research Center for Information Technologies at the University of Karlsruhe. His research interests include knowledge discovery in data and text, ontology engineering, learning and application of ontologies, and the Semantic Web. He received a diploma in industrial engineering and a PhD in applied informatics, both from the University of Karlsruhe. Contact him at FZI, Univ. of Karlsruhe, 76131 Karlsruhe, maedche@fzi.de; www.fzi.de/wim.

Steffen Staab is an assistant professor at the University of Karlsruhe and cofounder of Ontoprise GmbH. His research interests include computational linguistics, text mining, knowledge management, ontologies, and the Semantic Web. He earned a MSE from the University of Pennsylvania and he has received a Dr. rer. nat. from the University of Freiburg with his thesis on "Grading Knowledge - Extracting Degree Information from Texts" (LNAI 1744, 1999). In December 2001, he has submitted his habilitation about "Knowledge Management with Ontologies and Metadata" at the University of Karlsruhe. Contact him at the Institute AIFB, Univ. of Karlsruhe, 76128 Karlsruhe, Germany; sst@aifb.uni-karlsruhe.de

Contact:
Universität Karlsruhe (TH)
Institut AIFB
Andreas Hotho
D-76128 Karlsruhe
http://www.aifb.uni-karlsruhe.de/WBS/
hotho@aifb.uni-karlsruhe.de